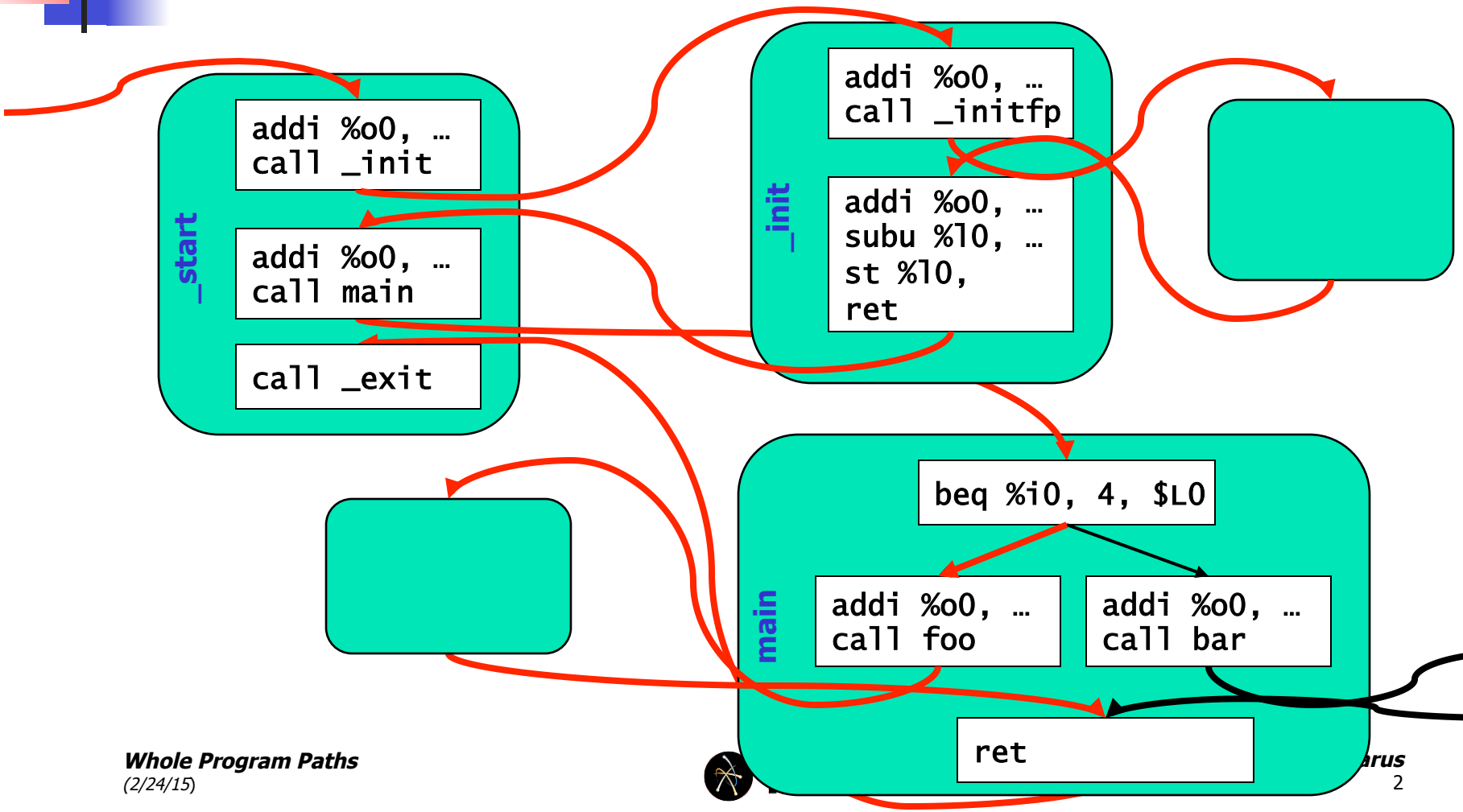
Whole Program Paths

James R. Larus
larus@microsoft.com

Whole Program Path



Example: 130.li

```
NODE *xlygetvalue(NODE *sym)
{
    register NODE *fp,*ep;
    for (fp = xlenv; fp; fp = cdr(fp))
      for (ep = car(fp); ep; ep = cdr(ep))
        if (sym == car(car(ep)))
          return (cdr(car(ep)));
    return (getvalue(sym));
}
```

- Most frequent path (10-20, 100K): 549,233 times
  - 56,021,766 instructions (6%)

Example: 130.li

```
NODE *xlygetvalue(NODE *sym)
{
    register NODE *fp,*ep;
    for (fp = xlenv; fp; fp = cdr(fp))
      for (ep = car(fp); ep; ep = cdr(ep))
        if (sym == car(car(ep)))
          return (cdr(car(ep)));
    return (getvalue(sym));
}
```

Example: 130.li

```
NODE *xlygetvalue(NODE *sym)
{
    register NODE *fp,*ep;
    for (fp = xlenv; fp; fp = cdr(fp))
      for (ep = car(fp); ep; ep = cdr(ep))
        if (sym == car(car(ep)))
          return (cdr(car(ep)));
    return (getvalue(sym));
}
```

Example: 130.li

```
NODE *xlygetvalue(NODE *sym)
{
    register NODE *fp,*ep;
    for (fp = xlenv; fp; fp = cdr(fp))
      for (ep = car(fp); ep; ep = cdr(ep))
        if (sym == car(car(ep)))
          return (cdr(car(ep)));
    return (getvalue(sym));
}
```

Example: 130.li

```
int xlobgetvalue(NODE *sym,NODE **pval)
{
    NODE *obj,*cls,*names;
    int ivtotal,n;
    obj = xlygetvalue(self);
    cls = xlygetvalue(msgclass);
    if (!(objectp(obj) && objectp(cls)))
        return (FALSE);
    for (; objectp(cls); cls = getivar(cls,SUPERCLASS)) {
        ...
        }
    }
    return (FALSE);
}
```

Example: 130.li

```
NODE *xlygetvalue(NODE *sym)
{
    register NODE *fp,*ep;
    for (fp = xlenv; fp; fp = cdr(fp))
      for (ep = car(fp); ep; ep = cdr(ep))
        if (sym == car(car(ep)))
          return (cdr(car(ep)));
    return (getvalue(sym));
}
```

Example: 130.li

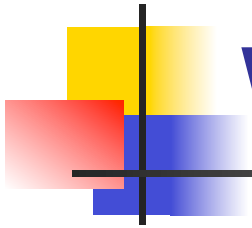
```
NODE *xlygetvalue(NODE *sym)
{
    register NODE *fp,*ep;
    for (fp = xlenv; fp; fp = cdr(fp))
      for (ep = car(fp); ep; ep = cdr(ep))
        if (sym == car(car(ep)))
          return (cdr(car(ep)));
    return (getvalue(sym));
}
```

Example: 130.li

```
NODE *xlygetvalue(NODE *sym)
{
     register NODE *fp,*ep;
     for (fp = xlenv; fp; fp = cdr(fp))
       for (ep = car(fp); ep; ep = cdr(ep))
         if (sym == car(car(ep)))
           return (cdr(car(ep)));
     return (getvalue(sym));
}
```

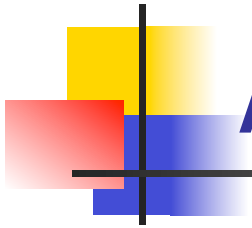
Example: 130.li

```
int xlobgetvalue(NODE *sym,NODE **pval)
{
    NODE *obj,*cls,*names;
    int ivtotal,n;
    obj = xlygetvalue(self);
    cls = xlygetvalue(msgclass);
    if (!(objectp(obj) && objectp(cls)))
        return (FALSE);
    for (; objectp(cls); cls = getivar(cls,SUPERCLASS)) {
        ...
        }
    }
    return (FALSE);
}
```



WPPs (Whole Program Paths)

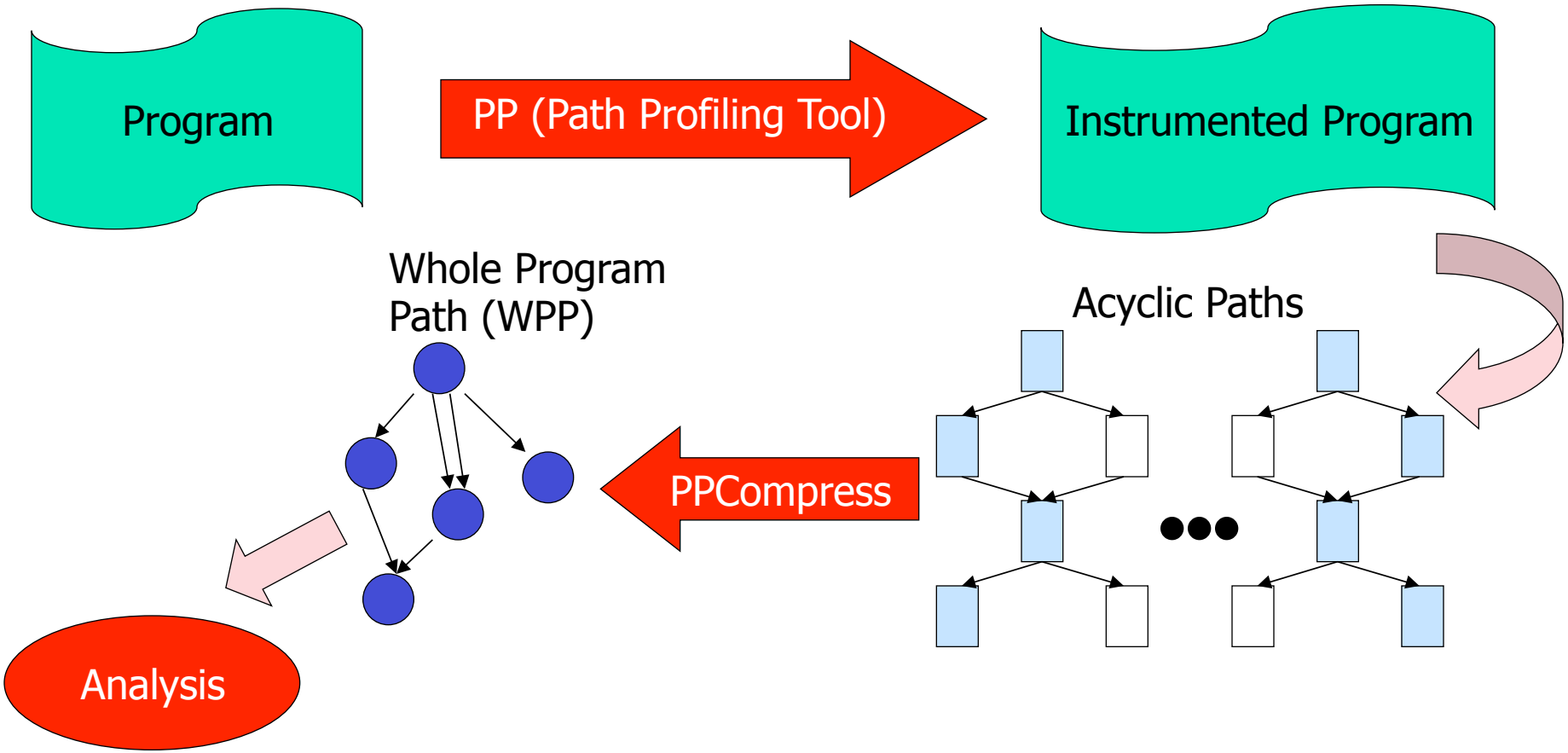
- Capture a program's dynamic control flow
- Like instruction traces, except:
  - WPP is far more compact
  - WPP's representation is directly analyzable
- Unlike profiles:
  - Dynamic sequencing



Applications

- Performance tuning
  - Fast paths
- Path-based compilation
- Trace-based computer architecture
- Error detection
  - Initialize; Use
  - Acquire; Use; Release

Whole Program Profiling



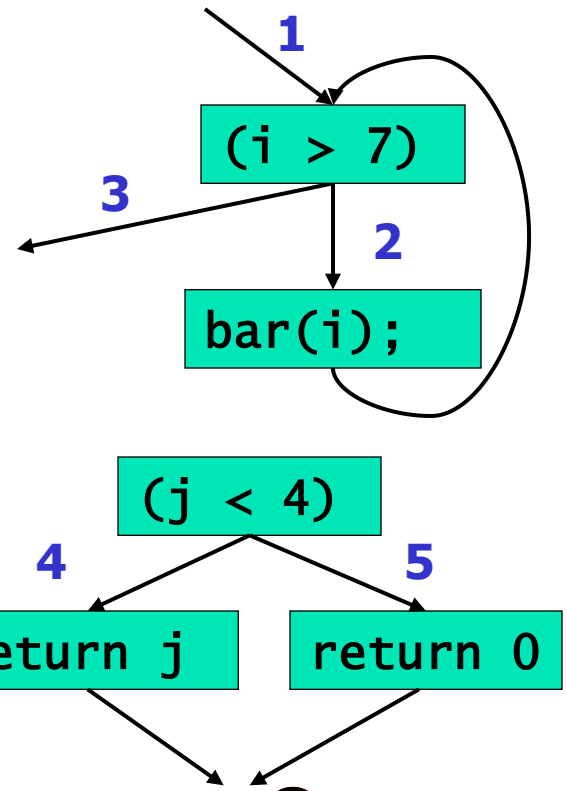
Overview – Run Time

Code

```
for (i=0; i<8; i++)
  bar(i);


int bar(int j)
{
  if (j < 4)
    return j;
  else
    return 0;
}
```

Acyclic Paths



Path Trace

```
1
4
2
4
2
4
2
4
2
5
2
5
2
5
2
5
3
```

Overview – Compression

Path Trace

1
4
2
4
2
4
2
4
2
5
2
5
2
5
2
5
3

Grammar

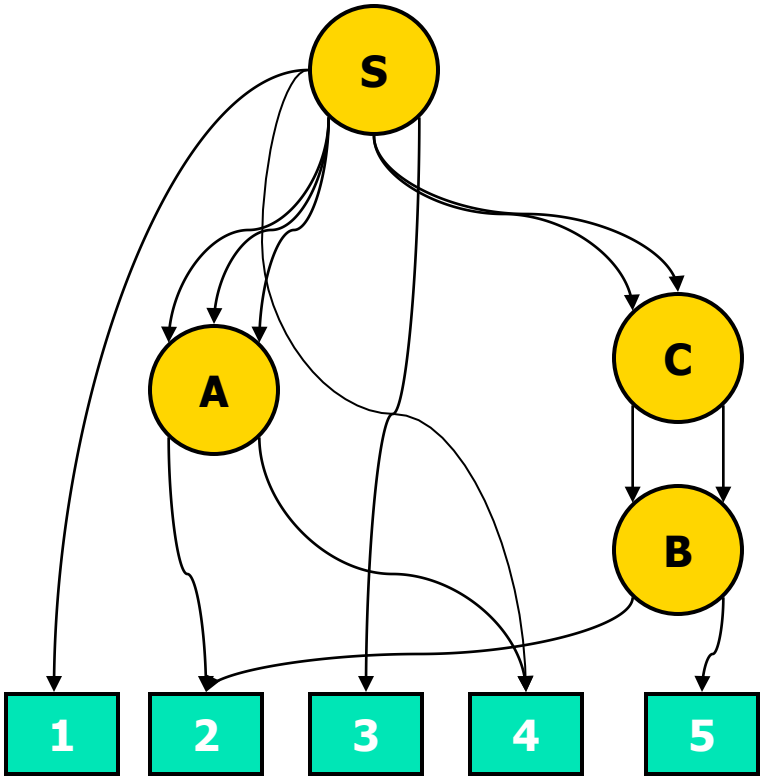
S ® 14AAACC3

A ® 24

B ® 25

C ® BB

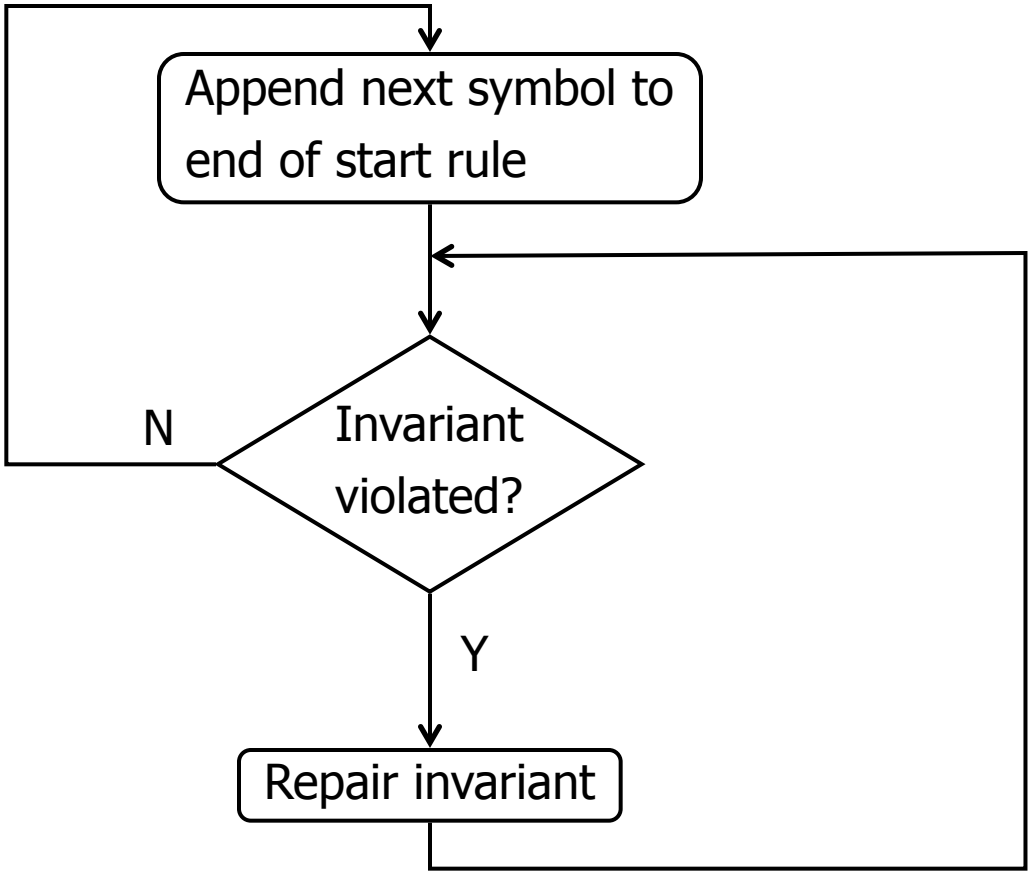
Whole Program Path (WPP)



SEQUITUR Compression

- ~Linear-time, on-line string compression
  - [Nevill-Manning & Witten ’97]
- Produces context-free grammar for string
- Insight: *log N* rules can produce *N* symbols
  - S ® AAB
    A ® BB ⇨ abcabcabcabcabc
    B ® abc

SEQUITUR Algorithm



Digram Uniqueness Invariant

- Digram is pair of consecutive symbols in rule
- All digrams occurs at most once in grammar rules
  - Correct by introducing/using rule

S ® abcab ⇨ S ® AcA
A ® ab

Rule Utility Invariant

- All non-terminals (except start symbol) must be referenced more than once
  - Correct by deleting rule

S ® BCBAd
A ® ab
B ® Ac
C ® Ad

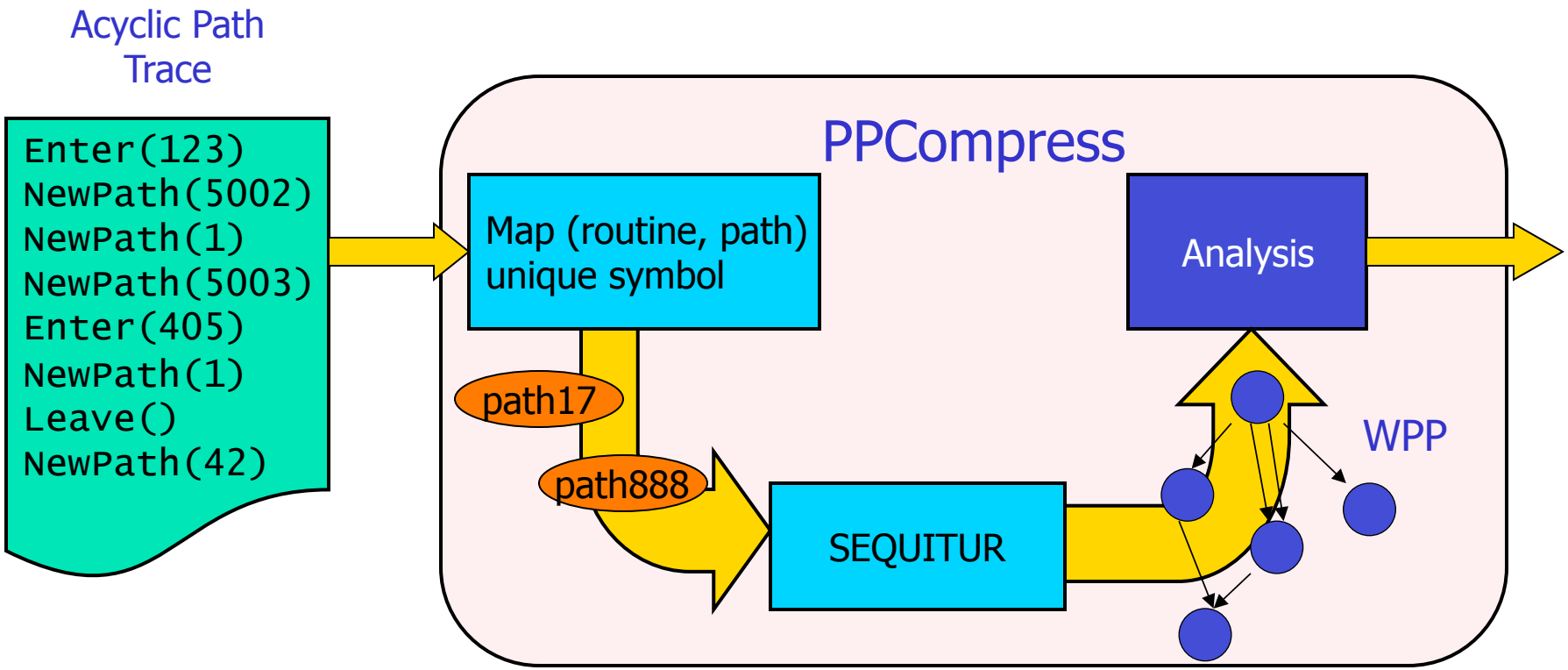
⇨

S ® DD
A ® ab
B ® Ac
C ® Ad
D ® BC

⇨

S ® DD
A ® ab
D ® AcAd

PPCompress


Acyclic Path Trace
Enter(123)
NewPath(5002)
NewPath(1)
NewPath(5003)
Enter(405)
NewPath(1)
Leave()
NewPath(42)
PPCompress
Map (routine, path) unique symbol
Analysis
path17
path888
SEQUITUR
WPP

WPPs

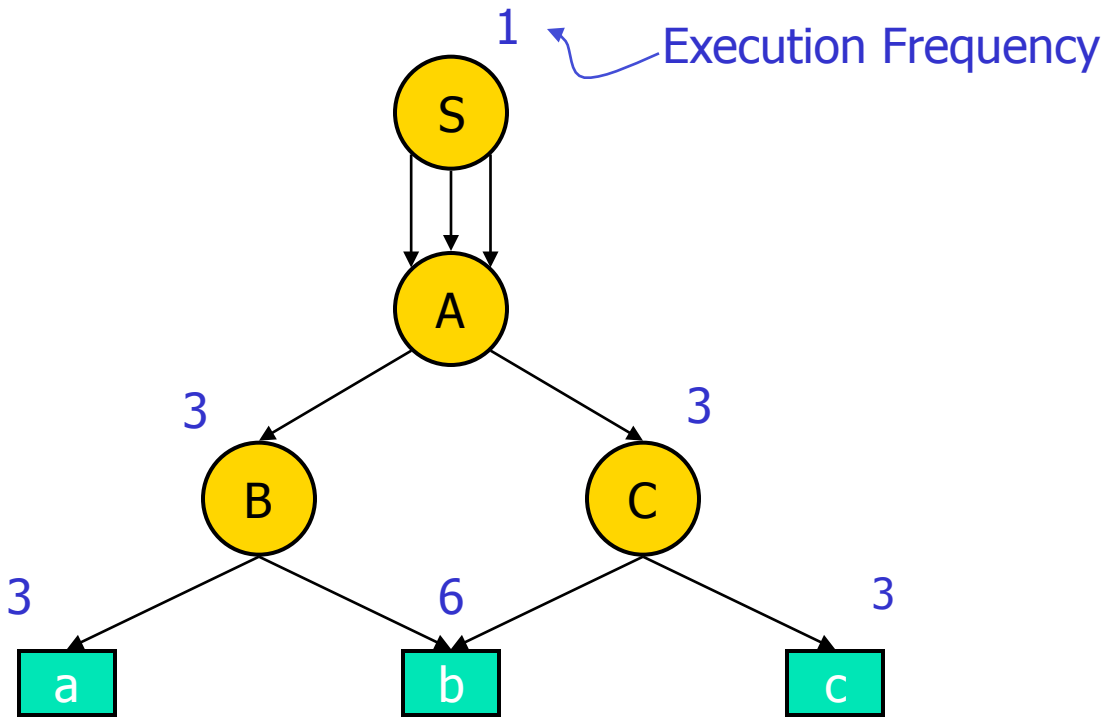
abbcabbcabbc

S → AAA

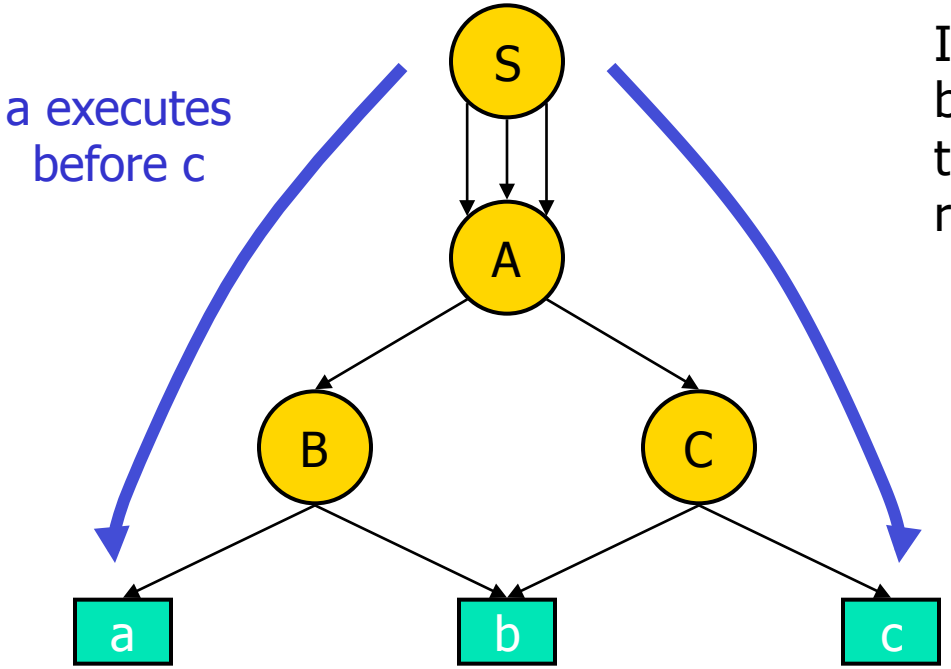
A → BC

B → ab

C ® bc

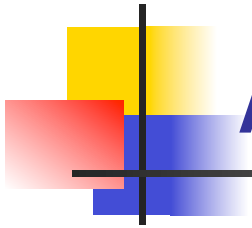


Execution Order



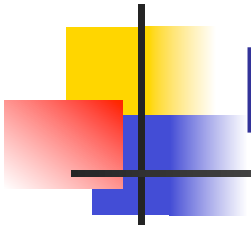
Is lock always acquired before running code that accesses shared resource?

Is memory always initialized after allocation and before use?



Analysis: Hot Paths

- Hot path is costly path through code
  - Heavily executed path
  - Costly operations along path
- Ammons, Ball, Larus [PLDI ’98]:
  - 40–99% of cost along 10–200 acyclic paths

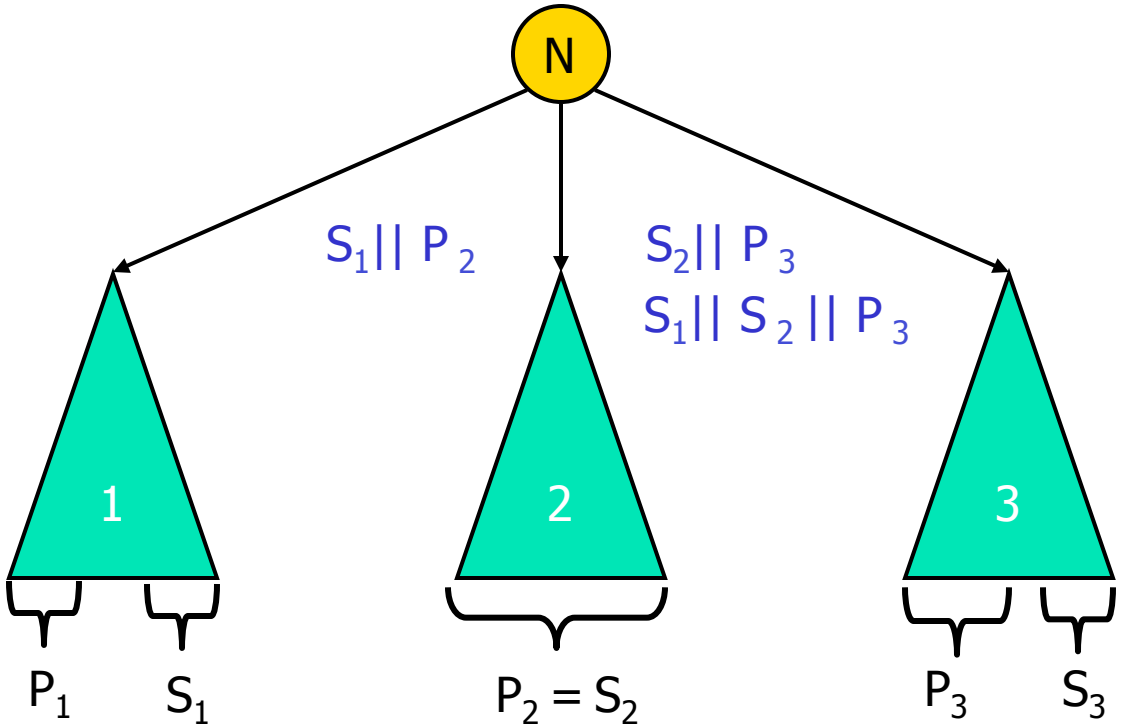


Hot Subpath

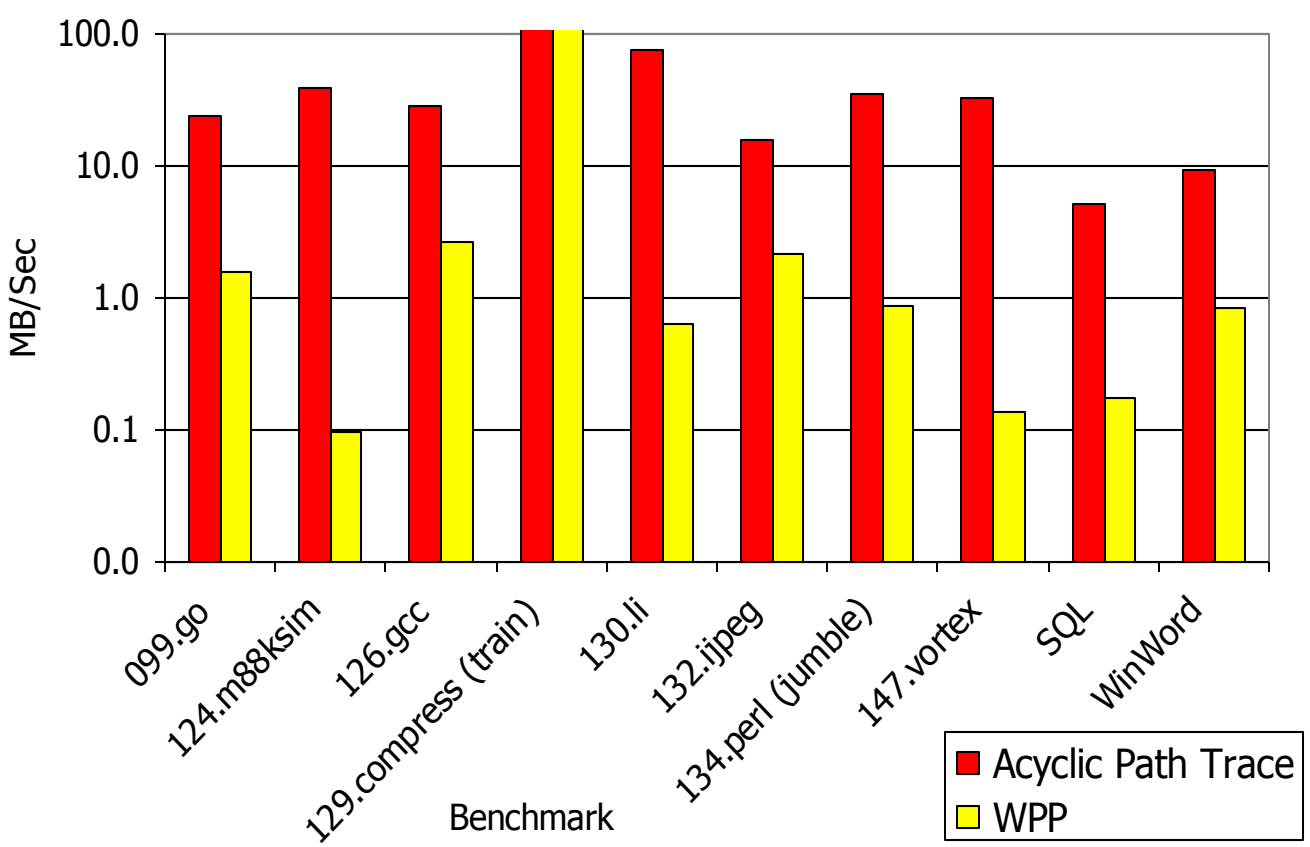
- Hot subpath (in WPP) lacks endpoints
  - Can always make it “hotter”
- Minimal hot subpath
  - Sequence of *L* or fewer consecutively executed acyclic paths
  - Cost of *C* or more

Finding Hot Subpaths

- Simple, linear algorithm finds hot subpaths



Data Production Rate

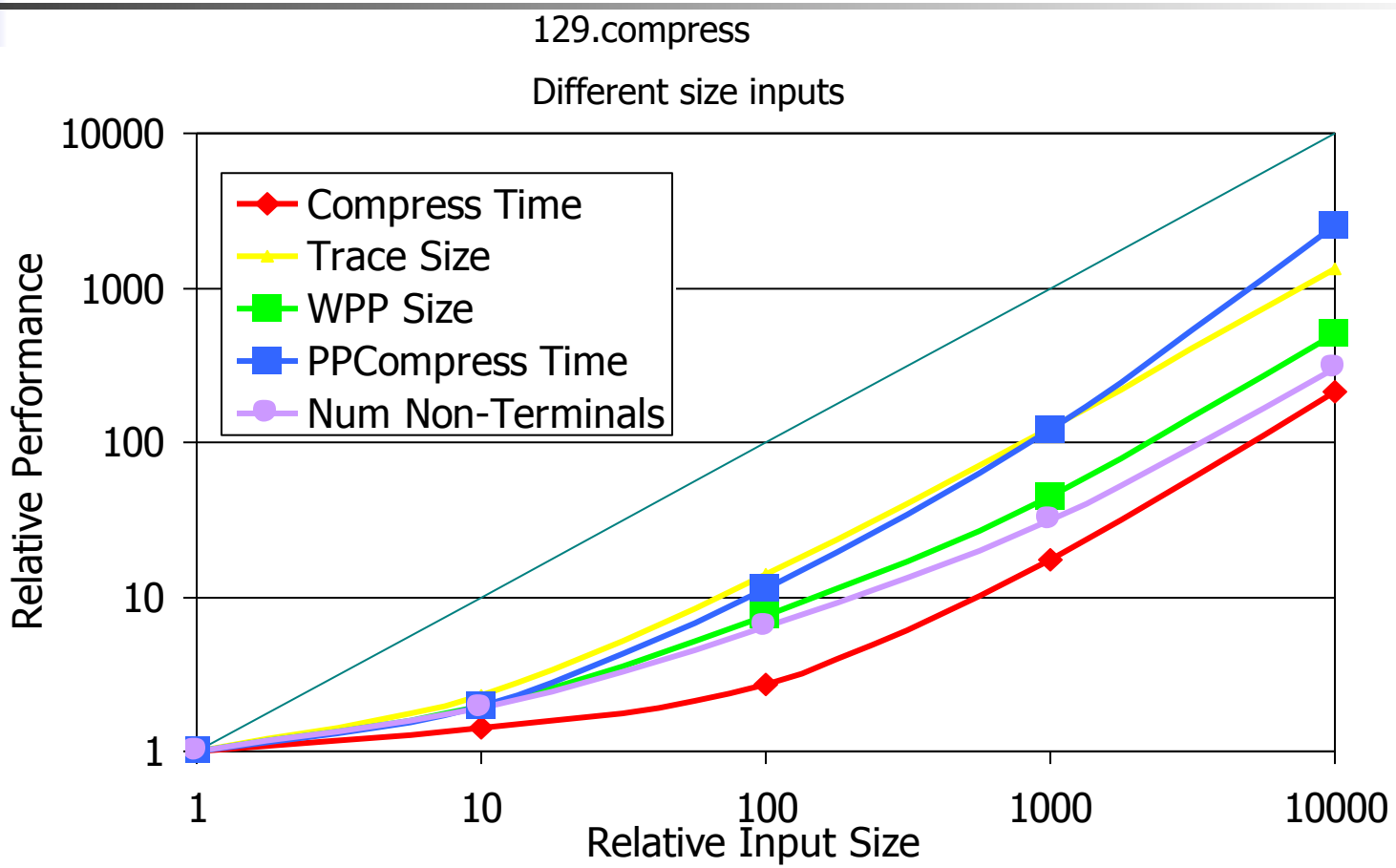


200 Mhz Pentium Pro

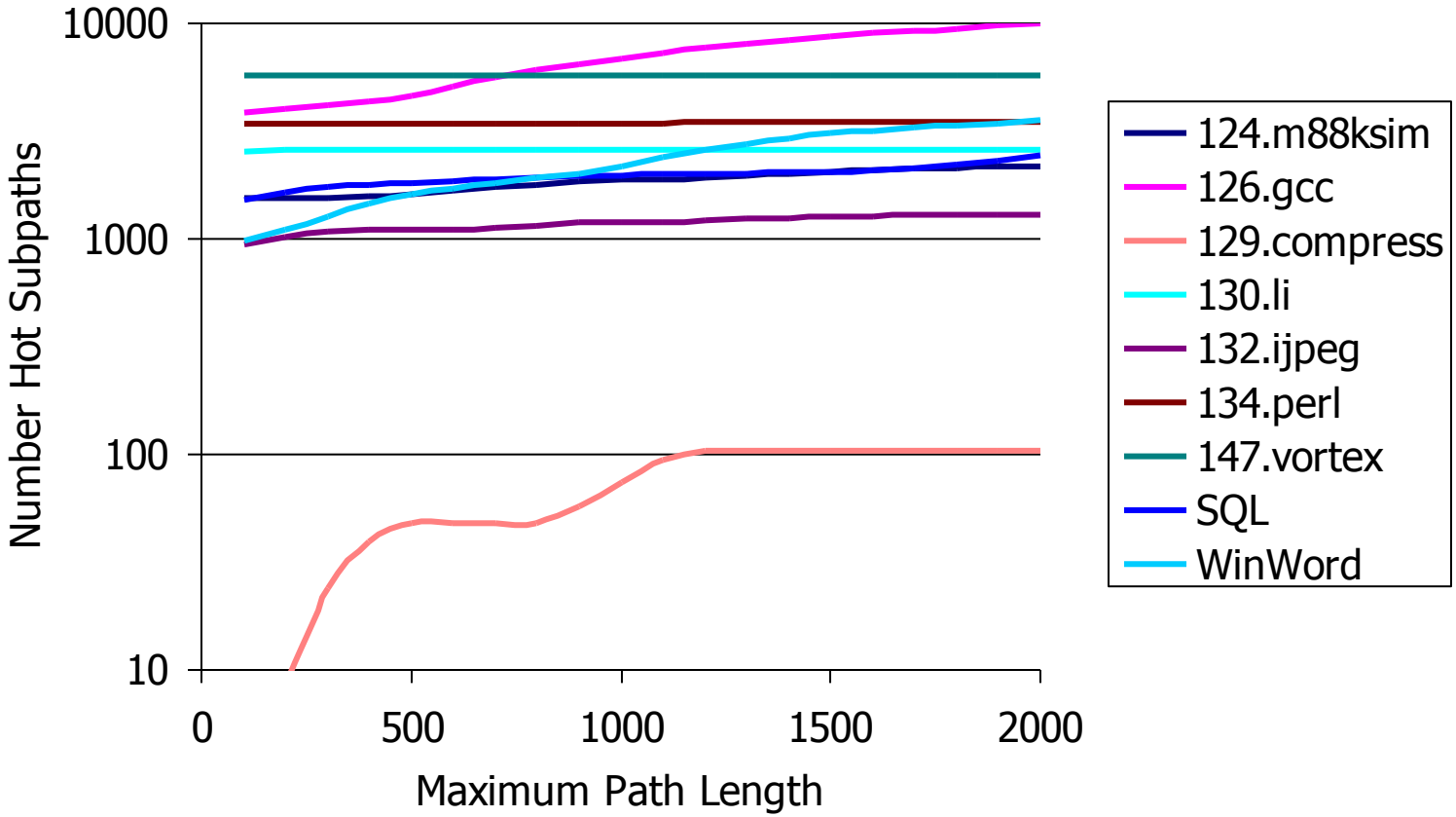
256 MB Memory

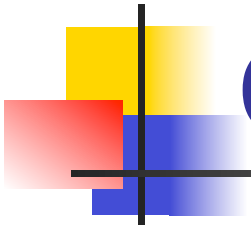
WinNT 4.0

PPCompress Performance



Hot Subpaths (Cost > 100K)





Conclusion

- Program's dynamic control flow can be captured and analyzed
  - WPP are practical for real applications
- Complements static program analysis
  - Complete info for one run vs. worst case estimate
  - Optimize for expected behavior